Intel® Pentium® M Processor on 90 nm Process with 2-MB L2 Cache

Specification Update

November 2005

Notice: The Intel® Pentium® M processor may contain design defects or errors known as errata which may cause the product to deviate from published specifications. Current characterized errata are documented in this Specification Update.

Document Number: 302209-015

Contents

Preface 5

Summary Tables of Changes 7

Identification Information 11

Errata 14

Specification Changes 23

Specification Clarifications 24

Documentation Changes 27

Revision History

Revision	Description	Date
-001	• Initial Release	May 2004
-002	• Updated Processor Identification Table	June 2004
-003	• Updated Processor Identification Table • Added Erratum X10	July 2004
-004	• Added Erratum X11, X12 and X13	October 2004
-005	• Added Erratum X14, X15 and X16	November 2004
-006	• Added Erratum X17 and X18	December 2004
-007	• Updated Processor Identification Table: Added C-0 S-Specs • Updated Summary Tables of Changes • Added Erratum X19 – X21	February 2005
-008	• Updated Summary of Tables of Changes • Updated Processor Identification Table • Added Specification Clarification X1	April 2005
-009	• Updated Processor Identification Table: Shading corrected from revision -008.	April 2005
-010	• Added Erratum X22 • Added Specification Clarification X2	May 2005
-011	• Updated Summary Tables of Changes • Added Erratum X23, X24 and X25	June 2005
-012	• Updated Processor Identification Table 1	July 2005
-013	• Updated Affected and Related Documents Tables • Updated Processor Identification Table 1 • Removed Erratum X13 (which was duplicate of X1)	July 2005
-014	• Added Specification Change X1	October 2005
-015	• Added Erratum X26	November 2005

§

Preface

This document is an update to the specifications contained in the documents listed in the following Affected Related Documents table. It is a compilation of device and document errata and specification clarifications and changes, and is intended for hardware system manufacturers and for software developers of applications, operating system, and tools.

Information types defined in the Nomenclature section of this document are consolidated into this update document and are no longer published in other documents. This document may also contain information that has not been previously published.

Affected Documents

Document Title	Document Number
The Intel® Pentium® M Processor on 90 nm Process with 2-MB L2 Cache Datasheet	302189-007
The Intel® Pentium® M Processor with 2-MB L2 Cache and 533-MHz Front Side Bus Datasheet	305262-002

Related Documents

Document Title	Document Number
IA-32 Intel® Architecture Software Developer's Manual, Volume 1: Basic Architecture	253665
IA-32 Intel® Architecture Software Developer's Manual, Volume 2A: Instruction Set Reference, A-M	253666
IA-32 Intel® Architecture Software Developer's Manual, Volume 2B: Instruction Set Reference, N-Z	253667
IA-32 Intel® Architecture Software Developer's Manual, Volume 3: System Programming Guide	253668
IA-32 Intel® Architecture Optimization Reference Manual	248966

Nomenclature

S-Spec Number is a five-digit code used to identify products. Products are differentiated by their unique characteristics, e.g., core speed, L2 cache size, package type, etc. as described in the processor identification information table. Care should be taken to read all notes associated with each S-Spec number.

Errata are design defects or errors. Errata may cause the behavior of the Intel® Pentium® M processor on 90 nm process with 2-MB L2 cache, to deviate from published specifications. Hardware and software, designed to be used with any given processor, must assume that all errata documented for that processor are present on all devices unless otherwise noted.

Documentation Changes include typos, errors, or omissions from the current published specifications. These changes will be incorporated in the next release of the specifications.

Specification Clarifications describe a specification in greater detail or further highlight a specification's impact to a complex design situation. These clarifications will be incorporated in the next release of the specifications.

Specification Changes are modifications to the current published specifications for the Pentium M processor. These changes will be incorporated in the next release of the specifications.

Summary Tables of Changes

The following table indicates the Specification Changes, Errata, Specification Clarifications or Documentation Changes, which apply to the listed MCH steppings. Intel intends to fix some of the errata in a future stepping of the component, and to account for the other outstanding issues through documentation or Specification Changes as noted. This table uses the following notations:

Codes Used in Summary Table

Stepping

X:	Erratum, Specification Change or Clarification that applies to this stepping.
(No mark) or (Blank Box):	This erratum is fixed in listed stepping or specification change does not apply to listed stepping.

Status

Doc:	Document change or update that will be implemented.
PlanFix:	This erratum may be fixed in a future stepping of the product.
Fixed:	This erratum has been previously fixed.
NoFix:	There are no plans to fix this erratum.

Row

Shaded:	This item is either new or modified from the previous version of the document.

Each specification update item is prefixed with a capital letter to distinguish the product. The key below details the letters that are used in Intel's microprocessor Specification Updates:

A = Intel® Pentium® II processor

B = Mobile Intel® Pentium® II processor

C = Intel® Celeron® processor

D = Intel® Pentium® II Xeon® processor

E = Intel® Pentium® III processor

F = Intel® Pentium® processor Extreme Edition and Intel® Pentium® D processor

G = Intel® Pentium® III Xeon® processor

H = Mobile Intel® Celeron® processor at 466 MHz, 433 MHz, 400 MHz, 366 MHz, 333 MHz, 300 MHz, and 266 MHz

J = 64-bit Intel® Xeon® processor MP with 1 MB L2 Cache

K = Mobile Intel® Pentium® III Processor – M

L = Intel® Celeron® D processor

M =Mobile Intel® Celeron® processor

N = Intel® Pentium® 4 processor

O = Intel® Xeon® processor MP

P = Intel® Xeon® processor

Q = Mobile Intel® Pentium® 4 processor supporting Hyper-Threading Technology† on 90 nm technology process

R = Intel® Pentium® 4 processor on 90 nm process

S = 64-bit Intel® Xeon® processor with 800 MHz system bus (1 MB and 2 MB L2 cache versions)

T = Mobile Intel® Pentium® 4 processor – M

U = 64-bit Intel® Xeon® processor MP with up to 8 MB L3 Cache

V = Mobile Intel® Celeron® processor on .13 Micron process in Micro-FCPGA Package

W = Intel® Celeron® M processor

X = Intel® Pentium® M processor on 90 nm process with 2-MB L2 Cache

Y = Intel® Pentium® M processor

Z = Mobile Intel® Pentium® 4 Processor with 533 MHz System Bus

Note: The specification updates for the Pentium processor, Pentium Pro processor, and other Intel products do not use this convention.

NO.	B1	C0	Plans	ERRATA
X1	X	X	No Fix	Code Segment (CS) Is Wrong on SMM Handler when SMBASE Is Not Aligned
X2	X	X	No Fix	IFU/BSU Deadlock May Cause System Hang
X3	X	X	No Fix	Memory Aliasing with Inconsistent A and D Bits May Cause Processor Deadlock
X4	X	X	No Fix	RDMSR or WRMSR to Invalid MSR Address May Not Cause GP Fault
X5	X	X	No Fix	Unable To Disable Reads/Writes to Performance Monitoring Related MSRs
X6	X	X	No Fix	Move to Control Register Instruction May Generate a Breakpoint Report
X7	X	X	No Fix	Error in Instruction Fetch Unit (IFU) Can Result in an Erroneous Machine Check-Exception (#MC)
X8	X	X	No Fix	Code Fetch Matching Disabled Debug Register May Cause Debug Exception
X9	X	X	No Fix	Upper Four PAT Entries Not Usable with Mode B or Mode C Paging
X10	X	X	No Fix	SSE/SSE2 Streaming Store Resulting in a Self-Modifying Code (SMC) Event May Cause Unexpected Behavior
X11	X	X	No Fix	Code Segment Limit Violation May Occur on 4 Gigabyte Limit Check
X12	X	X	No Fix	FST Instruction with Numeric and Null Segment Exceptions may cause General Protection Faults to be Missed and FP Linear Address (FLA) Mismatch
X13				Removed, see Erratum X1.
X14	X	X	No Fix	Page with PAT (Page Attribute Table) Set to USWC (Uncacheable Speculative Write Combine) While Associated MTRR (Memory Type Range Register) is UC (Uncacheable) May Consolidate to UC
X15	X	X	No Fix	Under Certain Conditions LTR (Load Task Register) Instruction May Result in System Hang
X16	X	X	No Fix	Loading from Memory Type USWC (Uncacheable Speculative Write Combine) May Get Its Data Internally Forwarded from a Previous Pending Store
X17	X	X	No Fix	FXSAVE after FNINIT without an Intervening FP (Floating Point) Instruction May Save Uninitialized Values for FDP (x87 FPU Instruction Operand (Data) Pointer Offset) and FDS (x87 FPU Instruction Operand (Data) Pointer Selector)
X18	X	X	No Fix	FSTP (Floating Point Store) Instruction under Certain Conditions May Result In Erroneously Setting a Valid Bit on an FP (Floating Point) Stack Register
X19		X	No Fix	An Execute Disable Bit Violation May Occur on a Data Page-Fault
X20		X	No Fix	CPUID Leaf 0x80000006 May Provide the Incorrect Value for an 8-Way Associative Cache
X21	X		PlanFix	Snoops during the Execution of a HLT (Halt) Instruction May Lead to Unexpected System Behavior
X22	X	X	No Fix	Invalid Entries in Page-Directory-Pointer-Table-Register (PDPTR) May Cause General Protection (#GP) Exception if the Reserved Bits are Set to One
X23	X	X	No Fix	INIT Does Not Clear Global Entries in the TLB

NO.	B1	C0	Plans	ERRATA
X24	X	X	No Fix	Use of Memory Aliasing with Inconsistent Memory Type May Cause System Hang
X25	X	X	No Fix	Machine Check Exception May Occur When Interleaving Code between Different Memory Types
X26	X	X	No Fix	Split I/O Writes Adjacent to Retry of APIC End of Interrupt (EOI) Request May Cause Livelock Condition

NO.	B1	C0	Plans	SPECIFICATION CLARIFICATION
X1	X	X	Doc	Specification Clarification with Respect to Time-stamp Counter
X2	X	X	Doc	Thermal Diode Offset Specification Clarification

Number	SPECIFICATION CHANGES
X1	AGTL+ Buffer On Resistance (Ron) Specification Correction

NO.	B1	C0	Plans	DOCUMENTATION CHANGES
				There are no Documentation Changes in this Specification Update revision.

§

Identification Information

The Intel Pentium M processor on 90 nm process with 2-MB L2 cache can be identified by the following values:

Family[1]	Model[2]	Brand ID[3]
0110	1101	00010110

NOTES:

1. The Family corresponds to bits [11:8] of the EDX register after Reset, bits [11:8] of the EAX register after the CPUID instruction is executed with a 1 in the EAX register.
2. The Model corresponds to bits [7:4] of the EDX register after Reset, bits [7:4] of the EAX register after the CPUID instruction is executed with a 1 in the EAX register.
3. The Brand ID corresponds to bits [7:0] of the EBX register after the CPUID instruction is executed with a1 in the EAX register.

Table 1. Identification Table

QDF/ S-spec	Processor Numbers	Product Stepping	FSB Frequency	CPU Signature	Core Speed		Package Micro-FCBGA-Pb= µ-BGA Lead Free	QDF Notes
					Highest Freq. Mode (HFM)	Lowest Freq. Mode (LFM)		
SL86G	730	C-0	533	06D8h	1.6 GHz	800 MHz	Micro-FCPGA	2,7
SL7SA	740	C-0	533	06D8h	1.73 GHz	800 MHz	Micro-FCPGA	2,7
SL7S9	750	C-0	533	06D8h	1.86 GHz	800 MHz	Micro-FCPGA	2,7
SL7SM	760	C-0	533	06D8h	2.0 GHz	800 MHz	Micro-FCPGA	2,7
SL7SL	770	C-0	533	06D8h	2.13 GHz	800 MHz	Micro-FCPGA	2,8
SL7VB	780	C-0	533	06D8h	2.26 GHz	800 MHz	Micro-FCPGA	9,2
SL86M	730	C-0	533	06D8h	1.6 GHz	800 MHz	Micro-FCBGA	2,7
SL7S8	740	C-0	533	06D8h	1.73 GHz	800 MHz	Micro-FCBGA	2,7
SL7SR	750	C-0	533	06D8h	1.86 GHz	800 MHz	Micro-FCBGA	2,7
SL7SQ	760	C-0	533	06D8h	2.0 GHz	800 MHz	Micro-FCBGA	2,7
SL7SP	770	C-0	533	06D8h	2.13 GHz	800 MHz	Micro-FCBGA	2,8
SL7SN	780	C-0	533	06D8h	2.26 GHz	800 MHz	Micro-FCBGA	9,2
SL86B	740	C-0	533	06D8h	1.73 GHz	800 MHz	Micro-FCBGA-Pb	2,7
SL86A	750	C-0	533	06D8h	1.86 GHz	800 MHz	Micro-FCBGA-Pb	2,7
SL869	760	C-0	533	06D8h	2.0 GHz	800 MHz	Micro-FCBGA-Pb	2,7
SL868	770	C-0	533	06D8h	2.13 GHz	800 MHz	Micro-FCBGA-Pb	2,8
SL8QK	780	C-0	533	06D8h	2.26 GHz	800 MHz	Micro-FCBGA-Pb	9,2
SL8QF	778	C-0	400	06D8h	1.6 GHz	600 MHz	Micro-FCBGA	2,3
SL89X	758	C-0	400	06D8h	1.5 GHz	600 MHz	Micro-FCBGA	2,3
SL8A3	723	C-0	400	06D8h	1.0 GHz	600 MHz	Micro-FCBGA	4,5
SL8LM	733J	C-0	400	06D8h	1.1 GHz	600 MHz	Micro-FCBGA	4,10

QDF/ S-spec	Processor Numbers	Product Stepping	FSB Frequency	CPU Signature	Core Speed		Package Micro-FCBGA-Pb= µ-BGA Lead Free	QDF Notes
					Highest Freq. Mode (HFM)	Lowest Freq. Mode (LFM)		
SL8A2	733J	C-0	400	06D8h	1.1 GHz	600 MHz	Micro-FCBGA	4,5
SL89Z	753	C-0	400	06D8h	1.2 GHz	600 MHz	Micro-FCBGA	4,5
SL8LL	753	C-0	400	06D8h	1.2 GHz	600 MHz	Micro-FCBGA	4,10
SL8QG	778	C-0	400	06D8h	1.6 GHz	600 MHz	Micro-FCBGA-Pb	2,3
SL89M	758	C-0	400	06D8h	1.5 GHz	600 MHz	Micro-FCBGA-Pb	2,3
SL89R	723	C-0	400	06D8h	1.0 GHz	600 MHz	Micro-FCBGA-Pb	2,3
SL8LT	733J	C-0	400	06D8h	1.1 GHz	600 MHz	Micro-FCBGA-Pb	4,10
SL89Q	733J	C-0	400	06D8h	1.1 GHz	600 MHz	Micro-FCBGA-Pb	4,5
SL89P	753	C-0	400	06D8h	1.2 GHz	600 MHz	Micro-FCBGA-Pb	4,5
SL8LS	753	C-0	400	06D8h	1.2 GHz	600 MHz	Micro-FCBGA-Pb	4,10
SL89N	738	C-0	400	06D8h	1.4 GHz	600 MHz	Micro-FCBGA-Pb	2,3
SL89Y	738	C-0	400	06D8h	1.4 GHz	600 MHz	Micro-FCPGA	2,3
SL7EM	755	B-1	400	06D6h	2.0 GHz	600 MHz	Micro-FCPGA	1,2
SL7EL	755	B-1	400	06D6h	2.0 GHz	600 MHz	Micro-FCBGA	1,2
SL7EN	745	B-1	400	06D6h	1.8 GHz	600 MHz	Micro-FCPGA	1,2
SL7EQ	745	B-1	400	06D6h	1.8 GHz	600 MHz	Micro-FCBGA	1,2
SL7EP	735	B-1	400	06D6h	1.7 GHz	600 MHz	Micro-FCPGA	1,2
SL7ER	735	B-1	400	06D6h	1.7 GHz	600 MHz	Micro-FCBGA	1,2
SL7EG	725	B-1	400	06D6h	1.6 GHz	600 MHz	Micro-FCPGA	1,2
SL7F2	725	B-1	400	06D6h	1.6 GHz	600 MHz	Micro-FCBGA	1,2
SL7GL	715	B-1	400	06D6h	1.5 GHz	600 MHz	Micro-FCPGA	1,2
SL7GK	715	B-1	400	06D6h	1.5 GHz	600 MHz	Micro-FCBGA	1,2
SL7F3	738	B-1	400	06D6h	1.4 GHz	600 MHz	Micro-FCBGA	2,3
SL7F4	733	B-1	400	06D6h	1.1 GHz	600 MHz	Micro-FCBGA	4,5
SL7VD	733	B-1	400	06D6h	1.1 GHz	600 MHz	Micro-FCBGA	4,5
SL7V2	723	B-1	400	06D6h	1.0 GHz	600 MHz	Micro-FCBGA	4,5

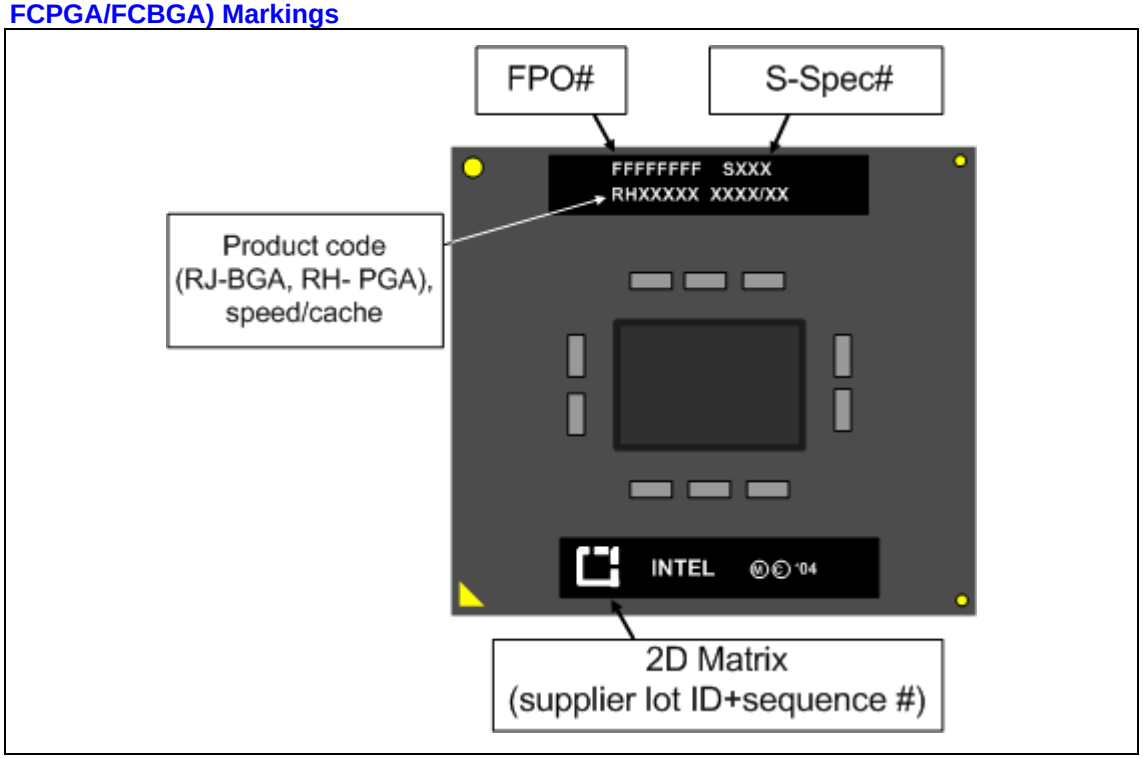
NOTES

1. Multiple HFM VID's; VCC_CORE = 1.340-1.276 V for Highest Frequency Mode (HFM).
2. VCC_CORE = 0.988 V for Lowest Frequency Mode (LFM).
3. VCC_CORE = 1.116 V for Highest Frequency Mode (HFM).
4. VCC_CORE = 0.812 V for Lowest Frequency Mode (LFM).
5. VCC_CORE = 0.940 V for Highest Frequency Mode (HFM).
6. Multiple HFM VID's; VCC_CORE = 1.356-1.308 V for Highest Frequency Mode (HFM)
7. VCC_CORE = 1.260-1.356 V for Highest Frequency Mode (HFM).
8. VCC_CORE = 1.260-1.372 V for Highest Frequency Mode (HFM).
9. VCC_CORE = 1.260-1.404 V for Highest Frequency Mode (HFM).
10. VCC_CORE = 0.876-0.956 V for Highest Frequency Mode (HFM).

Component Marking Information

The Intel Pentium M Processor on 90 nm process with 2-MB L2 cache may be identified by the following component markings:

Figure 1. Intel® Pentium® M Processor on 90 nm Process with 2-MB L2 Cache (Micro-FCPGA/FCBGA) Markings



§

Errata

X1. Code Segment (CS) Is Wrong on SMM Handler When SMBASE Is Not Aligned

Problem: With SMBASE being relocated to a non-aligned address, during SMM entry the CS can be improperly updated, which can lead to an incorrect SMM handler.

Implication: This is a rare condition that may result in a system hang. Intel has not observed this erratum with any commercially available software or system.

Workaround: Align SMBASE to 32 kB.

Status: For the steppings affected, see the *Summary of Tables of Changes*.

X2. IFU/BSU Deadlock May Cause System Hang

Problem: A lockable instruction with memory operand that spans across two pages may, given some rare internal conditions, hang the system.

Implication: When this erratum occurs, the system may hang. Intel has not observed this erratum with any commercially available software or system.

Workaround: Lockable data should always be contained in a single page.

Status: For the steppings affected, see the *Summary of Tables of Changes*.

X3. Memory Aliasing with Inconsistent A and D Bits May Cause Processor Deadlock

Problem: In the event that software implements memory aliasing by having two page directory entries (PDEs) point to a common page table entry (PTE) and the Accessed and Dirty bits for the two PDEs are allowed to become inconsistent the processor may become deadlocked.

Implication: This erratum has not been observed with commercially available software.

Workaround: Software that needs to implement memory aliasing in this way should manage the consistency of the Accessed and Dirty bits.

Status: For the steppings affected, see the *Summary of Tables of Changes*.

X4. RDMSR or WRMSR to Invalid MSR Address May Not Cause GP Fault

Problem: The RDMSR and WRMSR instructions allow reading or writing of MSR's (Model Specific Registers) based on the index number placed in ECX. The processor should reject access to any reserved or unimplemented MSRs by generating #GP(0). However, there are some invalid MSR addressers for which the processor will not generate #GP(0). This erratum has not been observed with commercially available software.

Implication: For RDMSR, undefined values will be read into EDX:EAX. For WRMSR, undefined processor behavior may result.

Workaround: Do not use invalid MSR addresses with RDMSR or WRMSR.

Status: For the steppings affected, see the *Summary of Tables of Changes.*

X5. Unable to Disable Reads/Writes to Performance Monitoring Related MSRs

Problem: The Performance Monitoring Available bit in the miscellaneous processor features MSR (IA32_MISC_ENABLES.7) was defined so that when it is cleared to a 0, RDMSR/ WRMSR/RDPMC instructions would return all zeros for reads of and prevent any write to Performance Monitoring related MSRs. Currently it is possible to read from or write to Performance Monitoring related MSRs when the Performance Monitoring Available bit is cleared to a 0.

Implication: It is not possible to disallow reads and writes to the Performance Monitoring MSRs. Intel has not observed this erratum with commercially available software of system.

Workaround: None.

Status: For the steppings affected, see the *Summary of Tables of Changes.*

X6. Move to Control Register Instruction May Generate a Breakpoint Report

Problem: A move (MOV) to Control register (CR) instruction where Control register is CR0, CR3 or CR4 may generate a breakpoint report.

Implication: MOV to Control Register Instruction is not expected to generate a breakpoint report.

Workaround: Ignore breakpoint data from MOV to CR instruction.

Status: For the steppings affected, see the *Summary of Tables of Changes.*

X7. Error in Instruction Fetch Unit (IFU) Can Result in an Erroneous Machine Check-Exception (#MC)

Problem: A rare combination of events including the generation of a bus lock(s), the execution of a WBINVD instruction, and a page accessed or dirty bit assist may result in an erroneous Machine Check-Exception (#MC).

Implication: Due to this erratum, unexpected machine check-exception (#MC) is generated. Intel has not been able to reproduce this erratum with commercially available software.

Workaround: It is possible for BIOS to contain a workaround for this erratum.

Status: For the steppings affected, see the *Summary of Tables of Changes.*

X8. Code Fetch Matching Disabled Debug Register May Cause Debug Exception

Problem: The bits L0-3 and G0-3 enable breakpoints local to a task and global to all tasks, respectively. If one of these bits is set, a breakpoint is enabled, corresponding to the addresses in the debug registers DR0-DR3. If at least one of these breakpoints is enabled, any of these registers are disabled (i.e., Ln and Gn are 0), and RWn for the disabled register is 00 (indicating a breakpoint on instruction execution), normally an instruction fetch will not cause an instruction-breakpoint fault based on a match with the address in the disabled register(s). However, if the address in a disabled register matches the address of a code fetch which also results in a page fault, an instruction-breakpoint fault will occur.

Implication: While debugging software, extraneous instruction-breakpoint faults may be encountered if breakpoint registers are not cleared when they are disabled. Debug software which does not implement a code breakpoint handler will fail, if this occurs. If a handler is present, the fault will be serviced. Mixing data and code may exacerbate this problem by allowing disabled data breakpoint registers to break on an instruction fetch.

Workaround: The debug handler should clear breakpoint registers before they become disabled.

Status: For the steppings affected, see the *Summary of Tables of Changes.*

X9. Upper Four PAT Entries Not Usable With Mode B or Mode C Paging

Problem: The Page Attribute Table (PAT) contains eight entries, which must all be initialized and considered when setting up memory types for the Pentium M processor. However, in Mode B or Mode C paging, the upper four entries do not function correctly for 4-Kbyte pages. Specifically, bit 7 of page table entries that translate addresses to 4-Kbyte pages should be used as the upper bit of a 3-bit index to determine the PAT entry that specifies the memory type for the page. When Mode B (CR4.PSE = 1) and/or Mode C (CR4.PAE) are enabled, the processor forces this bit to zero when determining the memory type regardless of the value in the page table entry. The upper four entries of the PAT function correctly for 2-Mbyte and 4-Mbyte large pages (specified by bit 12 of the page directory entry for those translations).

Implication: Only the lower four PAT entries are useful for 4-KB translations when Mode B or C paging is used. In Mode A paging (4-Kbyte pages only), all eight entries may be used. All eight entries may be used for large pages in Mode B or C paging.

Workaround: None.

Status: For the steppings affected, see the *Summary of Tables of Changes.*

X10. SSE/SSE2 Streaming Store Resulting in a Self-Modifying Code (SMC) Event May Cause Unexpected Behavior

Problem: An SSE or SSE2 streaming store that results in a Self-Modifying Code (SMC) event may cause unexpected behavior. The SMC event occurs on a full address match of code contained in L1 cache.

Implication: Due to this erratum, any of the following events may occur:

1. A data access break point may be incorrectly reported on the instruction pointer (IP) just before the store instruction.
2. A non-cacheable store can appear twice on the external bus (the first time it will write only 8 bytes, the second time it will write the entire 16 bytes).

Intel has not observed this erratum with any commercially available software. This erratum has been seen in a synthetic test environment.

Workaround: None.

Status: For the steppings affected, see the *Summary of Tables of Changes*.

X11. Code Segment Limit Violation May Occur on 4 Gigabyte Limit Check

Problem: Code Segment limit violation may occur on 4 Gigabyte limit check when the code stream wraps around in a way that one instruction ends at the last byte of the segment and the next instruction begins at 0x0.

Implication: This is a rare condition that may result in a system hang. Intel has not observed this erratum with any commercially available software, or system.

Workaround: Avoid code that wraps around segment limit.

Status: For the steppings affected, see the *Summary of Tables of Changes*.

X12. FST Instruction with Numeric and Null Segment Exceptions May Cause General Protection Faults to Be Missed and FP Linear Address (FLA) Mismatch

Problem: FST instruction combined with numeric and null segment exceptions may cause General Protection Faults to be missed and FP Linear Address (FLA) mismatch.

Implication: This is a rare condition that may result in a system hang. Intel has not observed this erratum with any commercially available software, or system.

Workaround: None identified.

Status: For the steppings affected, see the *Summary of Tables of Changes*.

X13. Removed; See Erratum X1.

X14. Page with PAT (Page Attribute Table) Set to USWC (Uncacheable Speculative Write Combine) While Associated MTRR (Memory Type Range Register) Is UC (Uncacheable) May Consolidate to UC

Problem: A page whose PAT memory type is USWC while the relevant MTRR memory type is UC, the consolidated memory type may be treated as UC (rather than WC as specified in *IA-32 Intel® Architecture Software Developer's Manua*l).

Implication: When this erratum occurs, the memory page may be as UC (rather than WC). This may have a negative performance impact.

Workaround: None identified.

Status: For the steppings affected, see the *Summary of Tables of Changes.*

X15. Under Certain Conditions LTR (Load Task Register) Instruction May Result in System Hang

Problem: An LTR instruction may result in a system hang if all the following conditions are met:

1. Invalid data selector of the TR (Task Register) resulting with either #GP (General Protection Fault) or #NP (Segment Not Present Fault).

2. GDT (Global Descriptor Table) is not 8-bytes aligned.

3. Data BP (breakpoint) is set on cache line containing the descriptor data.

Implication: This erratum may result in system hang if all conditions have been met. This erratum has not been observed in commercial operating systems or software. For performance reasons, GDT is typically aligned to 8-bytes.

Workaround: Software should align GDT to 8-bytes.

Status: For the steppings affected, see the *Summary of Tables of Changes.*

X16. Loading from Memory Type USWC (Uncacheable Speculative Write Combine) May Get Its Data Internally Forwarded from a Previous Pending Store

Problem: A load from memory type USWC may get its data internally forwarded from a pending store. As a result, the expected load may never be issued to the external bus.

Implication: When this erratum occurs, a USWC load request may be satisfied without being observed on the external bus. There are no known usage models where this behavior results in any negative side-effects.

Workaround: Do not use memory type USWC for memory that has read side-effects.

Status: For the steppings affected, see the Summary of Table of Change

X17. FXSAVE after FNINIT without an Intervening FP (Floating Point) Instruction May Save Uninitialized Values for FDP (x87 FPU Instruction Operand (Data) Pointer Offset) and FDS (x87 FPU Instruction Operand (Data) Pointer Selector)

Problem: An FXSAVE after FNINIT without an intervening FP instruction may save uninitialized values for FDP and FDS.

Implication: When this erratum occurs, the values for FDP/FDS in the FXSAVE structure may appear to be random values. These values will be initialized by the first FP instruction executed after the FXRSTOR that restore the saved floating point state. Any FP instruction with memory operand will initialize FDP/FDS. Intel has not observed this erratum with any commercially available software.

Workaround: After an FINIT, do not expect the FXSAVE memory image to be correct, until at least one FP instruction with a memory operand has been executed.

Status: For the steppings affected, see the *Summary of Tables of Changes.*

X18. FSTP (Floating Point Store) Instruction Under Certain Conditions May Result In Erroneously Setting a Valid Bit on an FP (Floating Point) Stack Register

Problem: An FSTP instruction with a PDE/PTE (Page Directory Entry/Page Table Entry) A/D bit update followed by user mode access fault due to a code fetch to a page that has supervisor only access permission may result in erroneously setting a valid bit of an FP stack register. The FP top of stack pointer is unchanged.

Implication: This erratum may cause an unexpected stack overflow.

Workaround: User mode code should not count on being able to recover from illegal accesses to memory regions protected with supervisor only access when using FP instructions.

Status: For the steppings affected, see the *Summary of Tables of Changes*.

X19. An Execute Disable Bit Violation May Occur on a Data Page-Fault

Problem: Under a combination of internal events, unexpected Execute Disable violations may occur on data accesses that are Execute Disable protected.

Implication: This erratum may cause unexpected Execute Disable violations.

Workaround: It is possible for the BIOS to contain a workaround for this erratum.

Status: For the steppings affected, see the *Summary of Tables of Changes*.

X20. CPUID Leaf 0x80000006 May Provide the Incorrect Value for an 8-Way Associative Cache

Problem: CPUID leaf 0x80000006 may return 0x8 in ECX [15:12] to indicate 8-way associative cache, but the correct encoding for an 8-way associative cache is 0x6.

Implication: Software that depends on the associativity of the cache may not function correctly.

Workaround: It is possible for the BIOS to contain a workaround for this erratum.

Status: For the steppings affected, see the *Summary of Tables of Changes*.

X21. Snoops during the Execution of a HLT (Halt) Instruction May Lead to Unexpected System Behavior

Problem: If during the execution of a HLT instruction an external snoop causes an eviction from the instruction fetch unit (IFU) instruction cache, the processor may, on exit from the HLT state, erroneously read stale data from the victim cache.

Implication: This erratum may lead to unexpected system behavior. Intel has only observed this condition in non-mobile configurations.

Workaround: It is possible for the BIOS to contain a workaround for this erratum.

Status: For the steppings affected, see the *Summary of Tables of Changes*.

X22. Invalid Entries in Page-Directory-Pointer-Table-Register (PDPTR) May Cause General Protection (#GP) Exception If the Reserved Bits are Set to One

Problem: Invalid entries in Page-Directory-Pointer-Table-Register (PDPTR) that have the reserved bits set to one, may cause a General Protection (#GP) exception.

Implication: Intel has not observed this erratum with any commercially available software.

Workaround: Do not set the reserved bits to one when PDPTR entries are invalid.

Status: For the steppings affected, see the *Summary of Table of Changes.*

X23. INIT Does Not Clear Global Entries in the TLB

Problem: INIT may not flush a TLB entry when:

1. The processor is in protected mode with paging enabled and the page global enable flag is set (PGE bit of CR4 register)
2. G bit for the page table entry is set
3. TLB entry is present in TLB when INIT occurs

Implication: Software may encounter unexpected page fault or incorrect address translation due to a TLB entry erroneously left in TLB after INIT.

Workaround: Write to CR3, CR4 or CR0 registers before writing to memory early in BIOS code to clear all the global entries from TLB.

Status: For the steppings affected, see the *Summary of Tables of Changes.*

X24. Use of Memory Aliasing with Inconsistent Memory Type May Cause System Hang

Problem: Software that implements memory aliasing by having more than one linear addresses mapped to the same physical page with different cache types may cause the system to hang. This would occur if one of the addresses is non-cacheable used in code segment and the other a cacheable address. If the cacheable address finds its way in instruction cache, and non-cacheable address is fetched in IFU, the processor may invalidate the non-cacheable address from the fetch unit. Any micro-architectural event that causes instruction restart will expect this instruction to still be in fetch unit and lack of it will cause system hang.

Implication: This erratum has not been observed with commercially available software.

Workaround: Although it is possible to have a single physical page mapped by two different linear addresses with different memory types, Intel has strongly discouraged this practice as it may lead to undefined results. Software that needs to implement memory aliasing should manage the memory type consistency.

Status: For the steppings affected, see the *Summary of Tables of Changes.*

X25. Machine Check Exception May Occur When Interleaving Code between Different Memory Types

Problem: A small window of opportunity exists where code fetches interleaved between different memory types may cause a machine check exception. A complex set of micro-architectural boundary conditions is required to expose this window.

Implication: Interleaved instruction fetches between different memory types may result in a machine check exception. The system may hang if machine check exceptions are disabled. Intel has not observed the occurrence of this erratum while running commercially available applications or operating systems.

Workaround: Software can avoid this erratum by placing a serializing instruction between code fetches between different memory types.

Status: For the steppings affected, see the *Summary of Tables of Changes.*

X26. Split I/O Writes Adjacent to Retry of APIC End of Interrupt (EOI) Request May Cause Livelock Condition

Problem: When Split I/O instruction writes occur adjacent to a retry of a Local APIC End of Interrupt (EOI) request by the chipset, a livelock condition may result. The required sequences of events are:

1. The processor issues a Local APIC EOI message.
2. The chipset responds with a retry because its downstream ports are full. It expects the processor to return with the same EOI request.
3. The processor issues a Split I/O write instruction instead.
4. The chipset responds with a retry because it expected the APIC EOI.
5. The processor insists the Split I/O write instruction must be completed and issues write instruction again.

Implication: A processor livelock may occur causing a system hang. This issue has only been observed in synthetic lab testing conditions and has not been seen in any commercially available applications. The erratum does not occur with Intel mobile chipset based platforms.

Workaround: Use the PIC instead of the APIC for the interrupt controller.

§

Specification Changes

X1. AGTL+ Buffer on Resistance (R_{ON}) Specification Correction

The AGTL+ Buffer On Resistance (Ron) specification has been corrected for *Intel Pentium M processor with 2-MB L2 cache and 533-MHz Front Side Bus. Intel® Pentium® M Processor with 2-MB L2 Cache and 533 MHz Front Side Bus Electrical, Mechanical, and Thermal Specification (EMTS)* Revision 2.1 Table 3-7 AGTL+ Signal Group DC Specifications shows R_{ON} specification as:

Symbol	Parameter	Min	Typ	Max	Unit
R_{ON}	Buffer On Resistance	22	25	28	Ω

Should state:

Symbol	Parameter	Min	Typ	Max	Unit
R_{ON}	Buffer On Resistance	17.7	24.7	32.9	Ω

§

Specification Clarifications

X1. Specification Clarification with Respect to Time-stamp Counter

In the "Debugging and Performance Monitoring" section (Sections 15.8, 15.10.9 and 15.10.9.3) of the *IA-32 Intel® Architecture Software Developer's Manual Volume 3: System Programming Guide*, the Time-stamp Counter definition has been updated to include support for the future processors. This change will be incorporated in the next revision of the *IA-32 Intel® Architecture Software Developer's Manual*.

15.8 Time-stamp Counter

The IA-32 architecture (beginning with the Pentium processor) defines a time-stamp counter mechanism that can be used to monitor and identify the relative time occurrence of processor events. The counter's architecture includes the following components:

a. **TSC flag —** A feature bit that indicates the availability of the time-stamp counter. The counter is available in an IA-32 processor implementation if the function CPUID.1:EDX.TSC[bit 4] = 1.

b. **IA32_TIME_STAMP_COUNTER MSR** (called TSC MSR in P6 family and Pentium processors) **—** The MSR used as the counter.

c. **RDTSC instruction —** An instruction used to read the time-stamp counter.

d. **TSD flag —** A control register flag is used to enable or disable the time-stamp counter (enabled if CR4.TSD[bit 2] = 1).

The time-stamp counter (as implemented in the P6 family, Pentium, Pentium M, Pentium 4, and Intel Xeon processors) is a 64-bit counter that is set to 0 following a RESET of the processor. Following a RESET, the counter will increment even when the processor is halted by the HLT instruction or the external STPCLK# pin. Note that the assertion of the external DPSLP# pin may cause the time-stamp counter to stop.

Members of the processor families increment the time-stamp counter differently:

e. For Pentium M processors (family [06H], models [09H, 0DH]); for Pentium 4 processors, Intel Xeon processors (family [0FH], models [00H, 01H, or 02H]); and for P6 family processors: the time-stamp counter increments with every internal processor clock cycle. The internal processor clock cycle is determined by the current core-clock to bus-clock ratio. Intel SpeedStep® technology transitions may also impact the processor clock.

f. For Pentium 4 processors, Intel Xeon processors (family [0FH], models [03H and higher]): the time-stamp counter increments at a constant rate. That rate may be set by the maximum core-clock to bus-clock ratio of the processor or may be set by the frequency at which the processor is booted. The specific processor configuration determines the behavior. Constant TSC behavior ensures that the duration of each clock tick is uniform and supports the use of the TSC as a wall clock timer even if the processor core changes frequency. This is the architectural behavior moving forward.

Note: To determine average processor clock frequency, Intel recommends the use of Performance Monitoring logic to count processor core clocks over the period of time for which the average is required. See Section 15.10.9 and Appendix A in this manual for more information.

The RDTSC instruction reads the time-stamp counter and is guaranteed to return a monotonically increasing unique value whenever executed, except for a 64-bit counter wraparound. Intel guarantees that the time-stamp counter will not wraparound within 10 years after being reset. The period for counter wrap is longer for Pentium 4, Intel Xeon, P6 family, and Pentium processors.

Normally, the RDTSC instruction can be executed by programs and procedures running at any privilege level and in virtual-8086 mode. The TSD flag allows use of this instruction to be restricted to programs and procedures running at privilege level 0. A secure operating system would set the TSD flag during system initialization to disable user access to the time-stamp counter. An operating system that disables user access to the time-stamp counter should emulate the instruction through a user-accessible programming interface.

The RDTSC instruction is not serializing or ordered with other instructions. It does not necessarily wait until all previous instructions have been executed before reading the counter. Similarly, subsequent instructions may begin execution before the RDTSC instruction operation is performed.

The RDMSR and WRMSR instructions read and write the time-stamp counter, treating the time-stamp counter as an ordinary MSR (address 10H). In the Pentium 4, Intel Xeon, and P6 family processors, all 64-bits of the time-stamp counter are read using RDMSR (just as with RDTSC). When WRMSR is used to write the time-stamp counter on processors before family [0FH], models [03H, 04H]: only the low order 32-bits of the time-stamp counter can be written (the high-order 32 bits are cleared to 0). For family [0FH], models [03H, 04H]: all 64 bits are writeable.

15.10.9 Counting Clocks

The count of cycles, also known as clockticks, forms a the basis for measuring how long a program takes to execute. Clockticks are also used as part of efficiency ratios like cycles per instruction (CPI). Processor clocks may stop ticking under circumstances like the following:

g. The processor is halted when there is nothing for the CPU to do. For example, the processor may halt to save power while the computer is servicing an I/O request. When Hyper-Threading Technology is enabled, both logical processors must be halted for performance-monitoring counters to be powered down.

h. The processor is asleep as a result of being halted or because of a power-management scheme. There are different levels of sleep. In the some deep sleep levels, the time-stamp counter stops counting.

There are three ways to count processor clock cycles to monitor performance. These are:

i. **Non-halted clockticks** — Measures clock cycles in which the specified logical processor is not halted and is not in any power-saving state. When Hyper-Threading Technology is enabled, this these ticks can be measured on a per-logical-processor basis.

j. **Non-sleep clockticks** — Measures clock cycles in which the specified physical processor is not in a sleep mode or in a power-saving state. These ticks cannot be measured on a logical-processor basis.

k. **Time-stamp counter** — Some processor models permit clock cycles to be measured when the physical processor is not in deep sleep (by using the time-stamp counter and the RDTSC instruction). Note that such ticks cannot be measured on a per-logical-processor basis. See Section 10.8 for detail on processor capabilities.

The first two methods use performance counters and can be set up to cause an interrupt upon overflow (for sampling). They may also be useful where it is easier for a tool to read a performance counter than to use a time-stamp counter (the timestamp counter is accessed using the RDTSC instruction).

For applications with a significant amount of I/O, there are two ratios of interest:

l. **Non-halted CPI** — Non-halted clockticks/instructions retired measures the CPI for phases where the CPU was being used. This ratio can be measured on a logical-processor basis when Hyper-Threading Technology is enabled.

Workaround: **Nominal CPI** — Time-stamp counter ticks/instructions retired measures the CPI over the duration of a program, including those periods when the machine halts while waiting for I/O.

15.10.9.3 Incrementing the Time-Stamp Counter

The time-stamp counter increments when the clock signal on the system bus is active and when the sleep pin is not asserted. The counter value can be read with the RDTSC instruction.

The time-stamp counter and the non-sleep clockticks count may not agree in all cases and for all processors. See Section 10.8 for more information on counter operation.

X2. Thermal Diode Offset Spec Clarification

The following text has been added to Intel® Pentium® M Processor on 90-nm Process with 2-MB L2 Cache and Intel® Pentium® M Processor on 90-nm Process with 2-MB L2 Cache and 533 MHz Front Side Bus datasheets chapter 5, section 5.1.2, Thermal Diode Offset:

The thermal diode offset model specific register, THERM_DIODE_OFFSET, is located at offset 03Fh accessed as a QWord. Bits 7:0 contain the thermal diode offset value in 0.5 °C resolution. This value should be subtracted from the diode measurement after the thermal diode ideality adjustments are made. Values from bits 7:0 of the MSR are interpreted as follows:

‘00000000’ = 0 °C

‘00000001’ = +0.5 °C

‘00000010’ = +1 °C

‘01111111’ = +63.5 °C

‘11111111’ = -0.5 °C

‘11111110’ = -1 °C

‘1000000’ = -64 °C

Appended to footnote of Table 5-7, *Thermal Diode Specification*:
Offset value is programmed in processor Model Specific Register 03Fh, “THERM_DIODE_OFFSET”

§

Documentation Changes

There are no Documentation Changes in this Specification Update revision.

§